Statistical Techniques in Robotics (16-831, S20) Lecture #15 (Wednesday, February 8)

MDP, Bellman Equations, Value Function

Lecturer: Kris Kitani *Scribes: Zhengyi Luo, Erica Weng*

1 Recap

In the previous lectures, we have been focusing on a series of online learning problems that are one-shot: each timestep of the experiment is independent and is not affected by the history or the future state/action of the environment. In this lecture, we will remove this assumption and study **sequential** learning problems where each action will affect the state of the environment and in turn affect the future state interaction between the agent and the environment. An example application is that of autonomous vehicle control. Given a sequence of car state readings, the goal is to control the car to achieve the maximum reward (or equivalently, the lowest cost) – that is, not crashing into obstacles and getting to the destination.

2 Reinforcement Learning

Supervised Learning	Reinforcement Learning
samples given by environment	samples (x, a) drawn through interaction with an environment, which behaves according to certain rules
samples (ζ, R) are assumed to be independently and identically distributed	samples within a "trajectory" ζ affect the next state (and thus the other relevant values in the next sample, such as reward R)
$x, a \sim \mathcal{D}(x, a)$	$\zeta, R \sim \mathcal{D}(\zeta, R)$
dataset is composed of N i.i.d datapoints: $\{(x_1, a_1), (x_2, a_2), \ldots, (x_N, a_N)\}$	dataset is composed of N trajectory - reward pairs: $\{(\zeta_1, R_1), (\zeta_2, R_2), \ldots, (\zeta_N, R_N)\}$ where $\zeta_i = \{(x_1, a_1), (x_2, a_2), \ldots, (x_T, a_T)\}$

Table 1: Comparison of Supervised Learning vs. Reinforcement Learning [1, 2, **?**]

Problem	Sampled	Evaluative	Sequential
PWEA	✗	✗	✗
OLC	✓	✗	✗
MAB	✗	✓	✗
C-MAB	✓	✓	✗
RL	✓	✓	✓

Table 2: Setup of the different problems we have learned so far

Reinforcement learning is a sampled problem because it's impossible to see the entire state space, and thus trajectories (data points) must be sampled. It's an evaluative problem in that we get qualitative feedback, but no information about optimality. And finally, it's a sequential problem, because the order we receive samples matters (within a trajectory), as they depend on each other across time.

3 Markov Decision Process

A Markov decision process is a discrete-time stochastic control process. [3] It can usually be described by a 4 tuple (S, A, P_a, R_a) where:

- S is a set of all possible states of the current environment ,
- A is a set of all possible actions of the current environment,
- $P_a(s_{t+1}|s_t, a_t)$ is the transition dynamics of the environment, and describes the probability that action a_t in state s_t will lead to s_{t+1},
- $R_a(s_t)$ or $R_a(s_t, a_t)$ or $R_a(s_t, a_t, s_{t+1})$ is the immediate reward recevied from the environment at time step t.

Other relevant definitions of the Markov decision process are:

- $\pi(a_t|s_t)$ is the policy that describes the probability of taking action a_t based on the current state s_t.
- $p_0(s_0)$ is the initial state probability, which describes the probability of the environment being initialized to state s_0.
- γ is the discount factor used to calculate the present value of the future reward.

The key assumption of a Markov decision process is the "Markov Property", which is defined as:

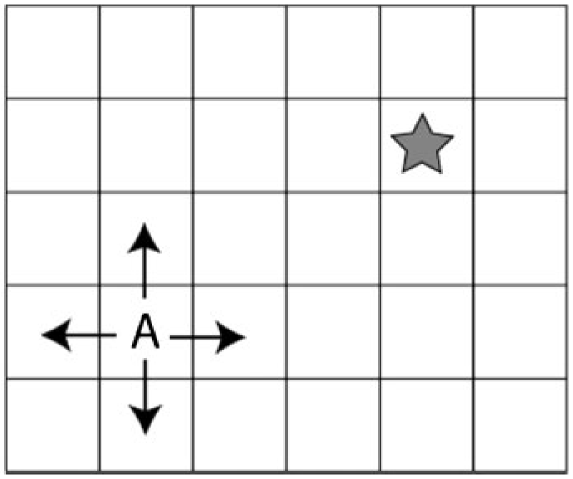
Markov Property: a state is Markov if and only if

$$P\left[S_{t+1} \mid S_t\right] = P\left[S_{t+1} \mid S_1, \ldots, S_t\right].$$

The above property states that a Markov decision process is "memory-less", where given the current state, the future state is independent of the past state. Namely, "the future is independent of the past given the present". Another way of stating this property is: the state S_t captures all relevant information from the history and can fully describe the current environment. Combining with the transition dynamics P_a, a policy π, and the Markov property, the joint probability of a Markov Decision Process $p\left(s_0, \cdots, s_T, a_0, \cdots, a_T\right)$ can be factorized as:

$$p\left(s_0, \cdots, s_T, a_0, \cdots, a_T\right) = p_0\left(s_0\right) \prod_{\mathrm{t}\,=\,0}^{T-1} P_a\left(s_{t+1} \mid s_t, a_t\right) \pi\left(a_t \mid s_t\right).$$

3.1 Example: Grid World



One of the simplest forms of a MDP is navigating in a grid world. Consider the above grid, where each square in the grid represents a state S; at each state, you can perform action a to move (up, down, left right). A policy $\pi(a_t|s_t)$ can be defined to take actions based on the current state, a reward $r(s_t)$ can be defined to map a state (block) to a real value, and a transition dynamics $P_a(s_{t+1}|s_t, a_t)$ describes the probability of transition to another state. The policy, reward, and transition dynamics can be either stochastic or deterministic.

4 Value Function

As mentioned before, the key difference between RL and online learning is the need to model **the future**. RL often models the future in the form of a value function. A value function estimates the total expected return of a trajectory. There are two major forms of value function: state value function $V(s_t)$ which models the total expected return of a trajectory starting in state s_t, and a action value function $Q(s_t, a_t)$ which models the total expected return of a trajectory starting in state s_t and performing action a_t. Since an expected trajectory needs to be defined with respect to a policy, we often denote the value functions as V^π and Q^π.

4.1 State Value Function

The state value function can be written out as:

$$V^\pi(s_t) = \mathbb{E}_p\left[r_t + r_{t+1} + r_{t+2} + \cdots \mid s_0 = s_t\right],$$

where the the expectation $\mathbb{E}_p$ is taken over the policy π and the transition dynamics P_a. The reward is defined as $r_t \triangleq r\left(s_{t+1}, a_t, s_t\right)$. The value function can be defined with respect to different time horizons, with:

$$\text{Infinite horizon return : } V^\pi(s_t) = \mathbb{E}_p\left[r_t + r_{t+1} + r_{t+2} + \cdots \mid s_0 = s_t\right],$$

$$\text{Finite horizon return: } V^\pi(s_t) = \mathbb{E}_p\left[r_t + r_{t+1} + r_{t+2} + \cdots r_T \mid s_0 = s_t\right],$$

$$\text{Infinite horizon: } V^\pi(s_t) = \mathbb{E}_p\left[\gamma^0 r_t + \gamma^1 r_{t+1} + \gamma^2 r_{t+2} + \cdots \mid s_0 = s_t\right].$$

4.2 State-Action Value Function

The state-action value function defines the total expected return of a trajectory starting in the state s_t and taking action a_t:

$$Q^{\pi}(s_t, a_t) = \mathbb{E}_p\left[\gamma^0 r\left(s_t\right) + \gamma^1 r\left(s_1\right) + \gamma^2 r\left(s_2\right) + \cdots \mid s_0 = s_t, a_0 = a_t\right].$$

Intuitively, the state-action value function, or "Q function" measures the future return based on taking the action a_t at state s_t.

Based on the definition of V^{π} and Q^{π}, we have the following relationship:

$$V^{\pi}(s_t) = \sum_{a_t} \pi(a_t \mid s_t) Q^{\pi}(s_t, a_t)$$

Proof (dropping the subscript for simplicity):

$$\begin{aligned}
V^{\pi}(s) &= \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_t \mid s_0 = s\right] \\
&= \sum_{s_{1:\infty}, a_{0:\infty}} p\left(s_{1:\infty}, a_{0:\infty}\right)\left[\sum_{t=0}^{\infty} \gamma^t r_t \mid s_0 = s\right] \\
&= \sum_{s_{1:\infty}, a_0:\infty} \pi\left(a_0 \mid s_0 = s\right) p\left(s_{1:\infty}, a_{1:\infty}\right)\left[\sum_{t=0}^{\infty} \gamma^t r_t \mid s_0 = s\right] \\
&= \sum_{a} \pi\left(a_0 = a \mid s_0 = s\right) \sum_{s_{1:\infty}, a_{1:\infty}} p\left(s_{1:\infty}, a_{1:\infty}\right)\left[\sum_{t=0}^{\infty} \gamma^t r_t \mid s_0 = s, a_0 = a\right] \\
&= \sum_{a} \pi\left(a_0 = a \mid s_0 = s\right) \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_t \mid s_0 = s, a_0 = a\right] \\
&= \sum_{a} \pi(a \mid s) Q^{\pi}(s, a)
\end{aligned}$$

References

[1] K. Kitani. 16-831 lecture slides.

[2] M. L. Littman. Reinforcement learning improves behaviour from evaluative feedback. *Nature*, 521(7553):445–451, 2015.

[3] R. Sutton and A. Barto. Reinforcement learning: An introduction. 27, 2018.